PATENT
44040-10001

# DEFORMABLE HEALER FOR REMOVING CAD DATABASE CONNECTIVITY GAPS FROM FREE-FORM CURVES AND SURFACES

## Cross-reference to Related Applications

**[0001]** This application claims priority from U.S. Provisional Patent Applications Serial No. 60/446,302, filed on February 11, 2003, entitled "Interactive Healer For Finding And Removing Cad Database Errors And Data Inefficiencies" to Celniker, and Serial No. 60/446,352, filed on February 11, 2003, entitled "Deformable Healer For Removing Cad Database Connectivity Gaps From Free-Form Curves And Surfaces" to Celniker, the contents of which are hereby incorporated by reference.

## Field of the Invention

**[0002]** The invention relates generally to computer-aided design (CAD) applications and to healing and interoperability applications that support transferring data between various CAD systems and specifically to modifying the shapes of free-form curves and surfaces contained in boundary representation (B-rep) solid models.

## Background of the Invention

**[0003]** The invention relates to the modification of shape functions represented in CAD databases to remove gap errors that impair the databases' ability to be translated between CAD systems and to be exploited by downstream CAD database applications without changing the intent of the original shape design documented by the CAD database.

**[0004]** CAD solid models containing free-form surfaces are typically represented as a network of patches across seam boundaries where each patch is itself a piece-wise continuous function [Farin 1988]. Often times, the patches cannot exactly connect together along their boundaries. There are small differences in the position and small variations in the cross-tangent and cross-curvature values along the seams creating gaps. These discontinuities are inherent to the geometry representations required for free-form

shapes. Typically, a CAD system will guarantee that these discontinuities will be limited to a specified tolerance value and it is now common to refer to these discontinuities as tolerances and managing the gaps between patches as the tolerancing problem.

**[0005]** Different CAD systems handle tolerances in different manners. Placing the tolerances of one CAD system on another often results in an unusable CAD model. The push for interoperability has been limited by a lack of an ability to explicitly manage free-form surface tolerances. Interoperability attempts at managing the tolerance problem have included finessing the issue by translating features rather than geometry [Rappaport 2003] and the use of common shared models [Bentley et al. 1998], and just ignoring the problem and translating the geometry directly via IGES, STEP, and specific point-to-point translators such as those marketed by Theorem Inc. However, features cannot generally represent free-form surfaces, and the many existing CAD systems and databases cannot take advantage of a common modeling approach. There is a need to translate free-form surfaces between CAD systems with a specified tolerance.

**[0006]** The need exists for a healing algorithm that will reduce the tolerances along the boundary of a free-form surface to a specified tolerance while minimizing any changes to its initial shape.

**[0007]** Mathematics and numerical methods for variational techniques for the interactive design of free-form shapes are known in the art. [Celniker 1992], [Celniker 1999] and [Celniker et al. 2002] apply variational techniques to the interactive design of free-form shapes but do not address the issue of enforcing boundary constraints that are not initially satisfied.

**[0008]** Enforcing constraints on geometry with non-variational approaches include the work of [Chyz 1985], [Gui 1993], [Lin 1981], [Pabon 1985], and [Rossignac 1988]. These approaches work with the parametric geometry representations such as line segments, planes, and conics, and do not extend to managing free-form surfaces. The theory of mathematical constraints is well known in the art. [Freidman I 1969], [Friedman II 1969], [Friedman III 1969].

## Summary of the Invention

**[0009]** This invention solves the problem of healing gap errors between free-form curves and surfaces and their bounding edges and vertices within a B-rep solid model by deforming the free-form shape to more closely approximate the locations of its bounding geometry while minimizing any changes to the original shape.

**[0010]** One aspect of the invention relates to an iterative method for refining a free-form shape consisting of a sequence of cycles where each cycle consists of checking the shape's current gap sizes and if larger than a specified tolerance optimizing that shape using the shape's current degrees of freedom to minimize the shape's gap vectors while also minimizing the shape's deviation from its initial position, and if the gaps remain larger than the given tolerance introducing new degrees of freedom to the shape by subdividing the shape's elements, and continuing this cycle until all gaps sizes are less than the given tolerance.

**[0011]** In a further aspect of the invention the optimization problem is stated as a variational problem with a cost function including a weighted penalty for changes in the free-form shape's initial position, stretch, and bending states, and a sum of weighted penalties for each gap distance. In alternative embodiments, the cost function includes a set of point constraints. The solution of the optimization problem finds the one shape out of all possible shapes that reduces the shape's gap sizes while minimizing any change to its original shape.

**[0012]** A further aspect of the invention relates to a heuristic method to refine the shape representation by choosing where to subdivide the shape elements based on the locations of gaps that exceed the given tolerance size.

**[0013]** A further aspect of the invention is a method for defining and finding a shape's gap vectors.

**[0014]** Another aspect of the invention relates to a method, called heal face-to-neighbors, for applying the curve and surface shape healing methods in a systematic manner to heal a target free-form face within a CAD B-rep solid model to its neighbor

faces. The method consisting of partitioning all the bounding edges of a target free-form surface into one of two groups: constrained or regenerated, and healing each constrained edge to its neighbor face, then healing the target face to its bounding vertices and constrained edges, and finally reconstructing all the regenerated edges with surface-surface intersections to form a free-form face in a solid model that has no gaps to any of it neighbor faces larger than a given tolerance.

## Brief Description of Drawings

**[0015]** FIG. 1 is a flow chart illustrating the iterative application of the Heal-face-to-neighbors method to all the free-form surface gap problems within a single CAD database.

**[0016]** FIG. 2 is a flow chart illustrating the steps of the Heal-face-to-neighbors method.

**[0017]** FIG. 3 is a flow chart illustrating the steps of the heal-edge-to-face Method.

**[0018]** FIG. 4 is a flow chart illustrating the deformable curve healing method.

**[0019]** FIG. 5 is a flow chart illustrating the deformable surface healing method.

**[0020]** FIGs. 6a-b illustrates a CAD database before and after the heal-face-to-neighbors method is applied.

## Description of Embodiments of the Invention

**[0021]** Deformable healing presents algorithms for healing errors in a B-rep CAD model that will reduce the tolerances along the boundary of a free-form surface to a specified tolerance while minimizing any changes to its initial shape. CAD models are produced by CAD systems such as CATIA, PRO-e, ACIS, or other CAD systems. The process of deformable healing states the tolerance problem as a variational optimization problem and solves it in a solid modeling context where changes to the geometry of faces, edges, and vertices have to be coordinated to preserve their geometric

relationships. This coordination requires the extension of the deformable healing algorithm to free-form curves in which the tolerances between a curve and a surface are minimized while minimizing the change in the curve's original shape. The process of deformable healing uses the mathematics and numerical methods disclosed in U.S. Patent No. 6,369,815 to Celniker et al [Celniker et al], the contents of which are herein incorporated by reference.

**[0022]** In one embodiment of the invention, the invention features an iterative heal-free-form-surfaces method, described in more detail below, which systematically removes or reduces to a specified tolerance all free-form gap problems from a CAD database by applying the heal-face-to-neighbors method described below to every free-form surface in a CAD database with bad gap problems.

**[0023]** The heal-face-to-neighbors method, described in more detail below, removes or reduces to a specified tolerance all the free-form gap problems associated with one free-form face in a solid modeling CAD database. It does this by systematically applying a heal vertex to neighbor method to each vertex connected to the face, applying a heal-edge-to-face method to every edge connected to the face, and finally by applying a deformable surface healing method, described in more detail below, to the surface itself.

**[0024]** The heal-edge-to-face method, described in more detail below, removes or reduces to a specified tolerance all the free-form gap problems associated with one free-form curve in a solid modeling CAD database. It does this by applying a deformable curve healing method to the curve itself. In this aspect the invention uses the method outlined below to heal the boundary of a CAD database face to its neighbors.

1. Moving vertices,
2. Heal-edge-to-face,
3. Deformable surface healing,
4. Rebuild edge by re-intersecting two surfaces, and
5. Rebuild vertex by re-intersecting faces.

The heal-face-to-neighbors aspect of the invention can enforce G0, G1, or G2 continuity between one face and its neighboring faces.

**[0025]** The deformable curve healing method described in more detail below minimizes the gaps between a curve and a target face and interpolates a set of target point locations while minimizing any change to the curve's initial shape. It does this by stating and solving a minimization problem to compute a new shape for the curve.

**[0026]** The deformable surface healing method described in more detail below minimizes the gaps between a surface and a set of target curves and interpolates a set of target point locations while minimizing any change to the surface's initial shape. It does this by stating and solving a minimization problem to compute a new shape for the surface.

**[0027]** Although the embodiment of the invention shown in FIG. 1 uses all of the methods described herein, alternative embodiments only use one or some of the specified methods to perform specific healing operations. In addition, alternative embodiments of the present invention can be used to heal analytic surfaces instead of free-form surfaces by converting the analytic surfaces to free-form surfaces and then applying the healing methods of the present invention.

Heal-Free-Form-Surfaces

**[0028]** An embodiment of the heal-free-form-surfaces method for healing all gap problems for all free-form surfaces in a CAD database is shown in FIG. 1. The method starts by identifying whether the database contains any free-form faces that have not yet been healed (2).

**[0029]** If more free-form faces need to be healed, the next face is selected (4). Any bad connectivity gaps on that face are fixed by applying the heal-face-to-neighbors method (6) described in more detail below. Vertices and edges that are modified by the heal-face-to-neighbors method are marked (8). On subsequent iterations the heal-face-to-neighbors method skips those edges and vertices that have already been healed.

[0030] The method repeats the steps of the method until all the free-form faces in the database have been visited.

Heal-Face-To-Neighbors

[0031] An embodiment of the method of heal-face-to-neighbors is shown in FIG. 2. Its purpose is to modify a CAD database to remove any gap problems between a free-form face and all of its neighbors. The first step of the method is to select the face to be healed (10). The selection can be done interactively by a user or automatically as part of a larger healing function.

[0032] Once the face is selected all of its connecting edges are classified as tangent or non-tangent (12). A tangent edge connects two faces in a CAD database that are tangent to one another along the length of the curve. A non-tangent edge connects two faces that have no points of tangency along the length of the curves. Edges with both tangent and non-tangent points are subdivided into edges that are cleanly tangent or non-tangent. The test for tangency is based on a geometric test of matching pairs of surface normal vectors sampled along the length of the edge on both connecting faces. The face that connects to the selected face through a tangent edge is called a tangent face.

[0033] All end-point vertices of the tangent edges are gathered (16). Each vertex location is checked to see that it lies on all the faces connected to the selected face at that vertex. If it does not, the location of the vertex is placed at the best possible place with respect to the connecting faces while minimizing its move (18). The best possible place is the intersection of the connecting faces if it exists. Otherwise it is the geometric average of projections to the faces and face intersections that is closest to the original point's position.

[0034] Once the end-point vertex locations are selected, each tangent edge is forced to lie on its tangent face by the heal-edge-to-face method described below using the end-point locations as point constraints and the tangent face's surface shape as the target surface shape (20).

**[0035]** After all the tangent edges are healed to their tangent faces, the shape of the select face's surface is modified by the deformable surface healing method described below using all the end-point locations as point constraints, the tangent edges as given curves, and the cross-tangent values sampled from the tangent faces along the length of the tangent curves as the given cross-tangent values (22).

**[0036]** The shape of every non-tangent edge is recomputed by intersecting the healed surface's shape with each non-tangent face (24). The reintersection not only computes the non-tangent edge's new shape but also the applicable end-point vertex locations.

**[0037]** When the method is successful, the CAD database is modified by replacing the selected face's surface with the healed surface shape, replacing all the connecting edge's shapes with the newly computed edge shapes, and replacing all the connecting edges end-point vertex locations with the newly computed vertex locations (26).

**[0038]** FIG. 6 shows an example of the operation of the heal-face-to-neighbors method for a CAD model. FIG. 6a shows the profile of a gap error between a B-spline surface 64 and a cylinder surface 66. The two surfaces are intended to be tangent along their connecting edge 68. The gap error can be seen as the overhang of surface 64 compared to surface 66. The heal-face-to-neighbors method is applied to the surface 64 and the result is shown in FIG 6b. The B-spline surface has been replaced by the new surface 70 that is now tangent to the original cylindrical surface 72 while approximating the original B-spline surface shape shown in 64.

Heal-Edge-To-Face

**[0039]** An embodiment of the method of heal-edge-to-face is shown in FIG. 3. Its purpose is to modify a CAD database to remove an edge to face bad gap problem. The first step of the method is to select the edge to modify and the face that the edge is to approximate (28). This selection process can be done interactively by an end user or automatically as part of a larger healing function.

[0040] The selected edge is used to gather the edge's end-point vertices (30). The locations of the end-point vertices are checked against the shape of the target face. When the vertices are not on the face, new locations are computed for them (32) by projecting the vertex locations onto the face. The vertices can be labeled to force skipping this step.

[0041] The shape of the edge is modified by the deformable curve healing method, described in more detail below, using the end-point locations as curve point constraints and the shape of the selected face as the target surface shape.

[0042] When the deformable curve healing method is successful, the CAD database is modified replacing the original end-point vertex locations with the new computed locations and replacing the edge's curve shape with the output curve shape computed by the deformable curve healing method (36).

Deformable Curve Healing

[0043] An embodiment of the method of deformable curve healing is illustrated in FIG. 4. Its purpose is to build a new curve description that approximates the shape of a given curve representation while interpolating a set of given point locations. Alternative embodiments build a new curve description approximating the shape of a given surface.

[0044] The method's first step is to classify the input point locations to see if any of those locations should be used as end-point locations for the curve (38). The classification is based on the distance of the closest specified location to the current curve end-point locations. When the specified locations are close to the end-point they are used as the target end-point location. When no point locations are specified near the current curve end-point location, the input curve's end-point location is used as the target end-point location.

[0045] A new piece of geometry, the approximating curve shape, is created (40). The underlying representation type for the approximating curve may be different than the input curve type. The approximate curve shape is the solution to an optimization problem that forces the approximating curve to interpolate the input curve's end-points while minimizing the variation between the curve's and the approximating curves

positions and tangent values. This allows the method to be applied to any CAD curve representation while producing any type CAD curve representation as output. Additionally this allows the method to remove any over sampling and parameterization errors that may be contained within the input curve's representation.

**[0046]** The next step of the method builds an optimization problem with constraints, described in more detail below (42). When solved, the optimization problem forces the approximate curve to interpolate the specified point locations and the selected curve end-point locations and minimize the displacement from the original curve shape (44). Alternative embodiments minimize the displacement from an optionally supplied surface shape. The mathematical description of the optimization problem follows in the next sub-section.

**[0047]** After solving the optimization problem, the approximating curve is analyzed to determine the quality of the solution. Point gap sizes are computed as the distance between the approximating curve and the specified point location (46). The displacement gap size is the maximum gap size between the approximating curve and the curve's original shape or the maximum gap size between the approximating curve and the optional surface shape when the optional surface shape is specified.

**[0048]** The point and displacement gap sizes are analyzed to determine whether all gap sizes are below a specified tolerance value (48). If all point and displacement gap sizes are below the tolerance value, the process is finished and the output of the method is the newly created and formed approximating curve shape. Often times the approximate shape will not meet the gap tolerance requirement. In all cases the cause of this failure is having too few degrees of freedom in the approximating curve's underlying representation. When the gap sizes fail the tolerance test, the method inserts new degrees of freedom into the approximating curve's representation by splitting in half the elements of the curve that have bad gap values (50). The modified approximating curve representation is then used to recreate the optimization problem and repeat the steps of the method (42). The method continues to iterate until the gap size tolerance test (48) is passed.

Deformable Curve Healing Optimization Problem

**[0049]** The steps of creating the approximate curve shape (40) and building the optimization problem (42) are performed as follows. Curve shape is represented by a parametric function as,

$$\mathbf{c}(s) = [x(s)\ y(s)\ z(s)]^T,\ s \in [a\ b] \subset \boldsymbol{R}$$
**Equation 1**

Where **c** is the curve shape, s is the curve's domain space variable bounded by the values a and b, and x, y, z are the 3 space coordinate functions for **C**. We model an approximation to the curve shape **c** with the approximate curve shape **w**(s). We limit the coordinate functions for **w** to be of the form,

$$f(s) = \Sigma\ f_i\ \varphi_i(s),\ s \in [a\ b] \subset \boldsymbol{R}.$$
**Equation 2**

Where $f_i$ is a scalar weight on the basis function, $\varphi_i(s)$. The optimization problem that is built and solved in 42and **44** minimizes the cost function,

$$C_{curve} = \int_{\sigma} (\alpha(\dot{c}-\dot{w})^2 + \beta(\ddot{c}-\ddot{w})^2 + \gamma(c-w)^2)\, d\sigma \quad - \sum_i \lambda_i \left(w(s_i) - w_i^0\right)$$
**Equation 3**

Where $\lambda_i$ are lambda variables for the point constraints and $\mathbf{w}(s_i)$ are the points on the approximating curve being forced to interpolate the specified target points, $\mathbf{w}_i^0$. The curve locations, $s_i$, are selected by projecting the target points to their nearest locations on the curve. The $\alpha$, $\beta$, and $\gamma$ gains may be varied to tune the quality of the approximation curve.

**[0050]** When building an approximate curve **w** that modifies a curve shape to interpolate a set of points while minimizing variations from its initial shape as in **40**, the function **c**(s) is the original input curve. When building an approximation curve that approximates a given surface, **c**(s) is computed as the projection of the approximate curve points defined by the last iteration of the method onto the given surface.

Deformable Surface Healing

**[0051]** An embodiment of the method of deformable surface healing is shown in FIG. 5. Its purpose is to build a new surface description that approximates the shape of a given surface representation while interpolating a set of given point locations and approximating a set of given curve shapes. Optionally the method can also force the surface to approximate the cross-tangent or the cross-tangent and cross-curvature values along the length of the curve.

**[0052]** The method's first step is to build a new piece of geometry, the approximating surface (52). The underlying representation type for the approximating surface may be different than the input surface type. The approximate surface shape is the solution to an optimization problem that forces the approximate surface to interpolate the corner points of the input surface while minimizing variations in the surface's position and tangent values. This allows the method to be applied to any CAD surface representation while producing any type CAD surface representation as output. Additionally this allows the method to remove any over sampling and parameterization errors that may be contained within the input surface's representation. For performance reasons, building the approximate surface may be skipped.

**[0053]** The next step of the method builds an optimization problem with constraints, described in more detail below (54). The optimization problem is then solved (56). When solved, the optimization problem forces the approximate surface to:

(i) interpolate the specified point locations,

(ii). minimize displacements between the approximate surface and the specified curve shapes,

(iii) optionally minimize the variation between the specified cross-tangent values and the approximate surface's cross-tangent values along the length of the given curves,

(iv) or optionally minimize the variation between the specified cross-tangent and cross-curvature values and the approximate surface's cross-tangent and cross-curvature values along the length of the given curves, and

(v) minimize the variation in the approximate surfaces position and tangent values from the input surface's position and tangent values.

**[0054]** When building the optimization problem for the first time, the approximation surface's underlying representation is sub-divided to ensure that no more than one point-constraint is located in any one element.

**[0055]** After solving the optimization problem, the approximating surface is analyzed to determine the quality of the solution (58). Point gap sizes are computed as the distance between the approximating surface and the specified point location. Curve gap sizes are defined as the maximum gap size along the length of a given curve and the approximating surface. The displacement gap size is the maximum gap size between the approximating surface and the surface's original shape.

**[0056]** The point, curve, and displacement gap sizes are analyzed to determine whether all gap sizes are below a specified tolerance value (60). If all point, curve and displacement gap sizes are below the specified tolerance level, the process is finished and the output of the method is the newly created and formed approximating surface shape. Often times the approximate shape will not meet the gap tolerance requirements. In all cases the cause of this failure is having too few degrees of freedom in the approximating surface's underlying representation. When the gap sizes fail the tolerance test, the method inserts new degrees of freedom into the approximating surface's representation by splitting in half the elements of the surface that have bad gap values (62). The modified approximating surface representation is then used to recreate the optimization problem (54) and the steps of the method are repeated. The method continues to iterate until the gap size tolerance test (60) is passed.

Deformable Surface Healing Optimization Problem

**[0057]** The steps of creating the approximating surface (52) and building the optimization problem (56) are performed as follows. Surface shape is represented by a parametric function as,

$$s(u,v) = [x(u,v)\ y(u,v)\ z(u,v)]^T,$$
$$u \in [a\ b] \subset \boldsymbol{R},$$
$$v \in [c\ d] \subset \boldsymbol{R}$$

**Equation 4**

Where **s** is the surface shape, u and v are the surface's domain space variables bounded by the values a, b, c and d, and x, y, z are the 3 space coordinate functions for **s**. We model an approximation to the surface shape **s** with the approximate surface shape **w**(u,v). We limit the coordinate functions for **w** to be of the form,

$$f(u,v) = \Sigma\, f_i\, \varphi_i(u,v),$$
$$u \in [a\ b] \subset \boldsymbol{R},$$
$$v \in [c\ d] \subset \boldsymbol{R}.$$

**Equation 5**

Where $f_i$ is a scalar weight on the basis function, $\varphi_i(u,v)$.

**[0058]** The optimization problem that is built and solved in **54** and **56** minimizes the cost function,

$$C_{surface} =$$

$$\int_{\sigma} (\dot{\mathbf{Q}}^T \bar{\alpha} \dot{\mathbf{Q}} + \ddot{\mathbf{Q}}^T \bar{\beta} \ddot{\mathbf{Q}} + \gamma\, \mathbf{Q}^2)\, d\sigma$$

$$- \sum_i \lambda_i \left( \mathbf{w}(u_i, v_i) - \mathbf{w}_i^0 \right)$$

$$+ \sum_i k_{curvegap} \int_{\sigma} (\mathbf{w}(\mathbf{c}_i^s(s)) - \mathbf{c}_i(s))^2\, ds + \sum_i k_{crosstangent} \int_{\sigma} (\mathbf{w}_n(\mathbf{c}_i^s(s)) - \mathbf{w}_{ni}(s))^2\, ds$$

$$+ \sum_i k_{crosscurvature} \int_{\sigma} (\mathbf{w}_{nn}(\mathbf{c}_i^s(s)) - \mathbf{w}_{nni}(s))^2\, ds$$

**Equation 6**

$$\text{Where } \mathbf{Q} = (w - s),\quad \dot{\mathbf{Q}} = \begin{bmatrix} q_u \\ q_v \end{bmatrix},\quad \ddot{\mathbf{Q}} = \begin{bmatrix} q_{uu} \\ q_{vv} \\ q_{vu} \end{bmatrix},\quad \text{and } \bar{\alpha} = \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix}\ \bar{\beta} = \begin{bmatrix} \beta_{11} & & \\ & \beta 22 & \\ & & \beta_{12} \end{bmatrix}.$$

**Equation 7**

Where $\lambda_i$ are lambda variables for the point constraints and $\mathbf{w}(u_i,v_i)$ are the points on the surface selected to interpolate the given set of input points, $\mathbf{w}_i^0$. $\mathbf{c}_i(s)$ are the specified curve shapes, $\mathbf{c}_i^s(s)$ are the parametric projections of the $\mathbf{c}_i(s)$ curves onto the input surface, and $\mathbf{w}_{ni}$ and $\mathbf{w}_{nni}$ are the optional target cross-tangent first and second derivative values measured along the given $\mathbf{c}_i(s)$ curves. The values $k_{curve\ gap}$, $k_{cross\text{-}tangent}$, and $k_{cross\text{-}}$

$_{curvature}$ are gain parameters, which can be adjusted to vary the solutions response to conflicting target values. The surface locations, $(u_i,v_i)$, are selected by projecting the target points to their nearest locations on the surface. The $\alpha$, $\beta$, and $\gamma$ gains may be varied to tune the quality of the approximation surface.

Implementation

**[0059]** The invention may be implemented in hardware or software, or a combination of both. However, preferably, the invention is implemented in computer programs executing on programmable computers. Each program is preferably implemented in a high level language (such as C++, Java, or Lisp) to communicate with a computer system. However, the programs can be implemented in assembly or machine language, if desired. In any case the language may be compiled or interpreted, procedural or symbolic.

**[0060]** Each such computer program is preferably stored on a storage media or device (e.g., ROM or magnetic/optical disk or diskette) readable by a general or special purpose programmable computer, for configuring and operating the computer when the storage media or device is read by the computer to perform the procedures described herein. The inventive system may also be considered to be implemented as a computer-readable storage medium, configured with a computer program, where the storage medium so configured causes a computer to operate in a specific and predefined manner to perform the functions described herein.

**[0061]** The foregoing disclosure of embodiments of the present invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many variations and modifications of the embodiments described herein will be obvious to one of ordinary skill in the art in light of the above disclosures. The scope of the invention is to be defined only by the claims appended hereto, and by their equivalents.